REMARKS

Reconsideration of the above-identified patent application in view of the amendments above and the remarks following is respectfully requested.

Claims 1 and 3-28 are in this case. Claims 1 and 3-28 have been rejected under § 103(a). Independent claims 1, 12, 18, 20 and 21 have been amended. New claims 29-35 have been added.

The claims before the Examiner are directed toward a flash memory device for storing code to be executed by a processor. The device includes a flash memory, for storing the code, such that the code cannot be executed in place from the flash memory. The device also includes a volatile memory to which the code is copied for execution, a logic, separate from the processor, for doing the copying, and a bus via which the flash memory, the volatile memory and the logic communicate directly with each other. If the code is boot code, the device can be used to boot a system, that includes the processor, in response to a power-on signal.

§ 103(a) Rejections – Brown et al. '739 in view of Kakinuma et al. '349

The Examiner has rejected claims 1, 10, 12, 13, 16, 18, 20 and 21 under § 103(a) as unpatentable over Brown et al., US Patent No. 6,201,739 (henceforth, "Brown et al. '739") in view of Kakinuma et al., US Patent No. 5,640,349 (henceforth, "Kakinuma et al. '349"). The Examiner's rejection is respectfully traversed.

Brown et al. '739 teach a flash memory device with a suspend pin for suspending write and erase operations to allow a read operation to take priority over a write or erase operation already in progress. The problem addressed by their device, as described in column 3 lines 36-60, is that of simultaneous management of data

storage and direct execution of code on the same flash memory: data storage occasionally needs write and erase operations, which interfere with the immediate access to code that is needed by direct execution. They note the applicability of their device to prior art systems such as the system illustrated in Figure 3, in which code stored in flash memory **104** is first copied to volatile memory **102** and then executed directly by processor **100** from volatile memory **102**. They also note in passing (column 5 line 33) that, although the primary intended application of their invention is to directly executable flash memories, their invention could also be used with flash memories that are not directly executable, for example with NAND flash memories.

Brown et al. '739 are silent about how code is copied from flash memory **104** to volatile memory **102**. As best understood, the copying is done by processor **100**.

Kakinuma et al. '349 teach that a flash memory controller **2**, separate from a host computer **1**, is conventionally used to copy data from a flash memory **4** to host computer **1**. In the invention of Kakinuma et al. '349, a similar flash memory controller **2** is used to copy data from flash memories **20** and **21** to host computer **1**. In the context of the system illustrated in Figure 3 of Brown et al. '739, flash memory controller **2** would copy data from a flash memory to volatile memory **102** via bus **108** for direct execution by processor **100**, thereby functioning similarly to logic **42** of the present invention.

The crucial difference between the present invention and this combination of Brown et al. '739 and Kakinuma et al. '349 is best understood in reference to Figure 2. Figure 2 shows that logic **42**, flash memory **14** and volatile memory component (S-RAM) **40** communicate with each other via an internal bus **37**. Logic **42** moves code, that is to be executed by CPU **32**, directly from flash memory **14** to volatile memory component **40** via internal bus **37**. Then CPU **32** accesses the code for execution

indirectly, via bus **38** and port **12** in addition to bus **37**. This is quite different from the obvious combination of Brown et al. '739 and Kakinuma et al. '349, which would be to have logic **42** move the code to RAM **34** via bus **38** for execution by CPU **32**. In the context of Kakinuma et al. '349, it would be as though host computer **1** executed code in buffer memories **22** and **23**. In fact, Kakinuma et al. '349 use buffer memories **22** and **23** only to buffer data between flash memories **20** and **21** and host computer **1**. There is neither a hint nor a suggestion in Kakinuma et al. '349 that a processor of host computer **1** could execute code resident in buffer memories **22** and **23**.

While continuing to traverse the Examiner's rejections, Applicant has, in order to expedite the prosecution, chosen to amend independent claims 1, 12, 18, 20 and 21 in order to clarify and emphasize this crucial distinction between the device and system of the present invention and the obvious combination of the teachings of Brown et al. '739 and Kakinuma et al. '349.

Claim 1 has been amended to recite, in addition to the flash memory, the volatile memory component and the logic, a bus, separate from the bus with which the external processor communicates with the flash-based unit, that the logic uses to move the portion of the code that is to be executed to the volatile memory component from the flash memory. Support for this amendment is found in Figure 2, which shows internal bus **37**, that logic **42** uses to move code from flash memory **14** to volatile memory component **40**, separate from bus **38** that CPU **32** uses to communicate with flash-based unit **36**. In addition, an extra period at the end of claim 1 has been removed.

Claim 12 has been amended to recite a first bus via which the volatile memory component is in direct communication with the restricted non-volatile memory, and

also to recite a second bus, separate from the first bus, via which the code in the volatile memory component is presented to the CPU. Support for this amendment is found in Figure 2, which shows internal bus **37** via which volatile memory component **40** is in direct communication with flash memory **14**, and also shows bus **38**, separate from internal bus **37**, via which the code in volatile memory component **40** is presented to CPU **32**.

Claim 18 has been amended to recite, in the flash-based unit, a first bus via which the volatile memory receives code to be executed from the flash memory, and also to recite, separately from the flash-based unit, a second bus via which the processor receives the code. Support for this amendment is found in Figure 2, which shows internal bus **37**, in flash-based unit **36**, via which volatile memory component **40** receives code to be executed from flash memory **14**, and also shows bus **38** via which CPU **32** receives the code.

Claims 20 and 21 have been amended to recite, in addition to the flash memory, the volatile memory and the logic, a bus whereby at most only the flash memory, the volatile memory and the logic communicate directly. Support for these amendments is found in Figure 2, which shows only flash memory **14**, volatile memory component **40** and logic **42** communicating directly via internal bus **37**. In particular, CPU **32** communicates with volatile memory **40** only indirectly, via bus **38** and port **12** in addition to bus **37**. The amendments state that "at most" only the flash memory, the volatile memory and the logic communicate directly via the bus in order to include within the scope of the claims the option described on page 9 line 23 through page 10 line 2:

> Alternatively, internal bus **37** may optionally handle communication directly only between port **12**, S-RAM **40** and logic **42**. The latter component then communicates with flash memory **14**...

Amended independent claims 1, 12, 18, 20 and 21 now feature language which makes it absolutely clear that in the device and system of the present invention, the flash memory (or the restricted non-volatile memory), the -volatile memory component and the logic communicate directly with each other via a common bus while the external processor communicates indirectly with these components via a separate bus. Applicant believes that the amendment of the claims completely overcomes the Examiner's rejections on § 103(a) grounds.

With independent claims 1 and 12 allowable in their present form, it follows that claims10, 13 and 16, that depend therefrom, also are allowable.

To further distinguish the present invention from the combined teachings of Brown et al. '739 and Kakinuma et al. '349, new dependent claims 29 and 30 have been added. New claims 29 and 30 add, to claims 20 and 21 respectively, the additional limitation that the flash-based unit includes a port for providing to the external processor the portion of the code that is received by the volatile memory component. Support for new claims 29 and 30 is found in the specification in Figure 2, which shows port **12** via which CPU **32** is provided the code received by volatile memory component **40**.

New independent claims 31-35 have been added to distinguish the present invention, as recited heretofore in claims 1, 12, 18, 20 and 21, in a different way from the combined teachings of Brown et al. '739 and Kakinuma et al. '349. The limitation that has been added to new claims 31-35, relative to the respective corresponding old claims 1, 12, 18, 20 and 21, is that the logic moves the at least portion of code from the flash memory or from the restricted non-volatile memory to the volatile memory component upon receipt of a power-on signal. There is neither a hint nor a suggestion in Brown et al. '739 and Kakinuma et al. '349, either separately or together, of

moving code from a non-volatile memory to a volatile memory in response to a power-on signal. Support for the new limitation of new claims 31-35 is found in the specification on page 10 lines 10-14:

> When the "power-on" signal is received, indicating that system **30** should now "boot up", a busy signal on bus **38** signals CPU **32** not to begin operation (stage 1). Next, a specific code (set of one or more instructions) is copied automatically from flash memory **14** to S-RAM **40**, without the intervention of CPU **32** (stage 2).

§ 103(a) Rejections – Brown et al. '739 and Anderson et al. '577

The Examiner has rejected claims 3-5 under § 103(a) as being unpatentable over Brown et al. '739 and Anderson et al., US Patent No. 6,295,577. The Examiner's rejection is respectfully traversed.

It is demonstrated above that independent claim 1 is allowable in its present form. It follows that claims 3-5, that depend therefrom, also are allowable.

§ 103(a) Rejections – Brown et al. '739 an Mills et al. '688

The Examiner has rejected claims 6 and 7 under § 103(a) as being unpatentable over Brown et al. '739 and Mills et al., US Patent No. 6,385,688. The Examiner's rejection is respectfully traversed.

It is demonstrated above that independent claim 1 is allowable in its present form. It follows that claims 6 and 7, that depend therefrom, also are allowable.

§ 103(a) Rejections – Brown et al. '739 and Nakata '101

The Examiner has rejected claims 8, 9, 14 and 15 under § 103(a) as being unpatentable over Brown et al. '739 and Nakata, US Patent No. 6,523,101. The Examiner's rejection is respectfully traversed.

It is demonstrated above that independent claims 1 and 12 are allowable in their present form. It follows that claims 8, 9, 14 and 15, that depend therefrom, also are allowable.

§ 103(a) Rejections – Brown et al. '739 and Esfahani et al. '695

The Examiner has rejected claims 11, 17, 19, 22 and 23 under § 103(a) as being unpatentable over Brown et al. '739 and Esfahani et al., US Patent No. 6,434,695 (henceforth, "Esfahani et al. '695"). The Examiner's rejection is respectfully traversed.

It is demonstrated above that independent claims 1 and 12 are allowable in their present form. It follows that claims 11 and 17, that depend therefrom, also are allowable.

Esfahani et al. '695 describe what Apple Computer, Inc. did when the "Toolbox ROM" code of the OS code of its Macintosh computer became too big to fit into the computer's ROM. Only low-level OS code **31** is stored in Boot ROM **11** and is executed directly from Boot ROM **11**. The rest of the code that heretofore would have been stored in Boot ROM **11** now is stored elsewhere, in compressed form, and is copied to RAM **12** for execution. Some of low-level OS code **31**, specifically the RTAS, may be copied to RAM **12** and executed from RAM **12**, as stated in column 7 lines 2-4, but as best understood this code is executed from RAM **12** only after all the OS code has booted.

By contrast, according to the present invention as recited in independent claims 19 and 22, boot code for basic initialization of the system is executed from the volatile memory component by the processor to boot the system. There is neither a hint nor a suggestion in Esfahani et al. '695 of any utility to executing RTAS or any other component of low-level OS code **31** from RAM **12** in order to boot the OS code.

Furthermore, independent claims 19 and 22 state that the portion of the boot code that is transferred to the volatile memory component is the portion of the boot code for basic initialization of the system. As best understood, the corresponding portion of low-level OS code **31** of Esfahani et al. '695 is (column 6 lines 36-38)

> code for performing Power-On Self Test (POST), including code for performing diagnostics, generating a boot beep and an error beep

and not the RTAS. There is neither a hint nor a suggestion that this portion of low-level OS code **31** ever is copied to RAM **12** and executed from RAM **12**.

Thus, independent claims 19 and 22 are allowable in their present form over the prior art cited by the Examiner.

With independent claim 22 allowable in its present form, it follows that claim 23, that depends therefrom, also is allowable.

§ 103(a) Rejections - Brown et al. '739 and Esfahani et al. '695 and further in view of Mahmoud '911

The Examiner has rejected claims 24-28 under § 103(a) as being unpatentable over Brown et al. '739 and Esfahani et al. '695 and in further view of Mahmoud, US Patent No. 6,567,911 (henceforth, "Mahmoud '911"). The Examiner's rejection is respectfully traversed.

Mahmoud '911 teaches a method of booting a computer system whose system RAM has "an option ROM memory space of a confined size" (column 5 line 2). The Examiner has interpreted this "confined size" as meaning that the system RAM of Mahmoud '911 is only large enough to store the boot code that is executed therefrom, similar to the volatile memory component recited in claims 24-28 that is only large enough to store the portion of the boot code that is used for basic initialization of the system. In fact, system RAM **102** of Mahmoud '911, as illustrated in Figure 2A, is

much larger than the boot code that is executed therefrom. The "confined size" of the option ROM memory space is a confined size of the address space, from address C800:00 to address DFFF:00, that is used for the option ROM memory space. This address space, as best understood, is much smaller than the total address space of system RAM **102**. It is not at all obvious from Mahmoud '911 that any purpose would be served by providing a RAM that is only as large as the option ROM memory space of Mahmoud '911. Therefore, independent claims 24-28 are allowable in their present form over the prior art cited by the Examiner.

In view of the above amendments and remarks it is respectfully submitted that independent claims 1, 12, 18-22, 24-28 and 31-35, and hence dependent claims 3-11, 13-17, 23, 29 and 30 are in condition for allowance. Prompt notice of allowance is respectfully and earnestly solicited.

Respectfully submitted,

Mark M. Friedman
Attorney for Applicant
Registration No. 33,883

Date: November 17, 2003